I-7083/7083D/7083B/7083BD

User's Manual

I-7000 New Features
1. Self Tuner Inside
2. Multiple Baud Rate
3. Multiple Data Format
4. Dual WatchDog Inside
5. True Distributed Control
6. High Speed & High Density I/O

Your Powerful Tools → **Create New Ideas** → **Create New Applications**

Warranty

All products manufactured by ICP DAS are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICP DAS assume no liability for damages consequent to the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, no responsibility is assumed by ICP DAS for its use, nor for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Trademark

The names used for identification only maybe registered trademarks of their respective companies.

Table of Contents

1. Introduction

I-7000 is a family of network data acquisition and control modules. They provide A/D, D/A, DI/O, Timer/Counter and other functions. These modules can be remote controlled by a set of commands. The common features of I-7083/7083B are given as following:

- 3 axis, 32-bit encoder counter
- Encoder counting mode: Cw/Ccw , Pulse/Direction, A/B Phase
- Maximum counting rate: 1MHz
- Encoder Input: A, B, Z differential
- Input Level: 5V, 12V/24V with external resistor
- A/B/Z signal isolation voltage: 2500V optical isolation
- Built-in XOR logic for active high or active low encoder input

The I-7083B will save the counter value to EEPROM when the power goes off. The 7083D & 7083BD equip a 7-Seg interface to display encoder value & ABZ status one by one.

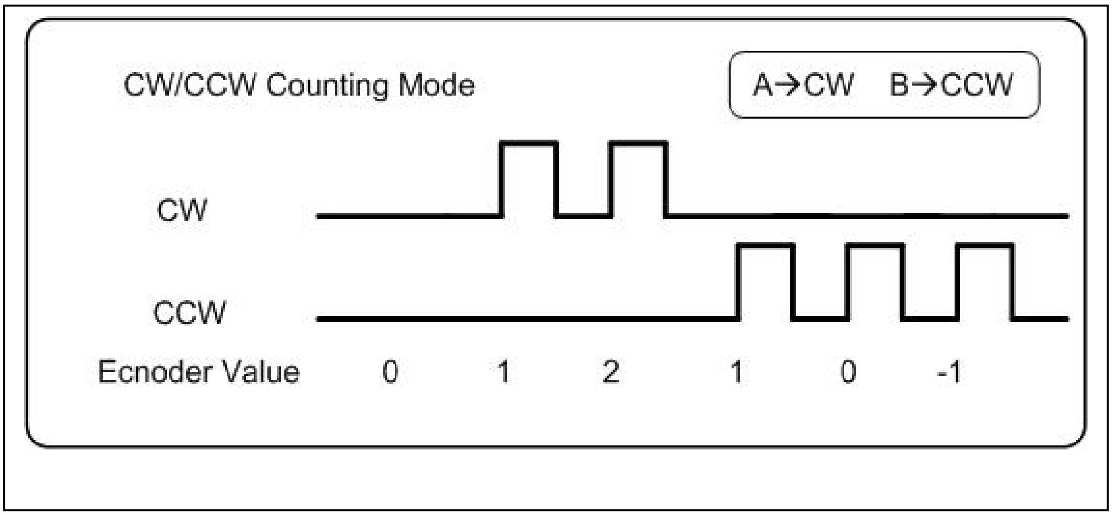
More Information

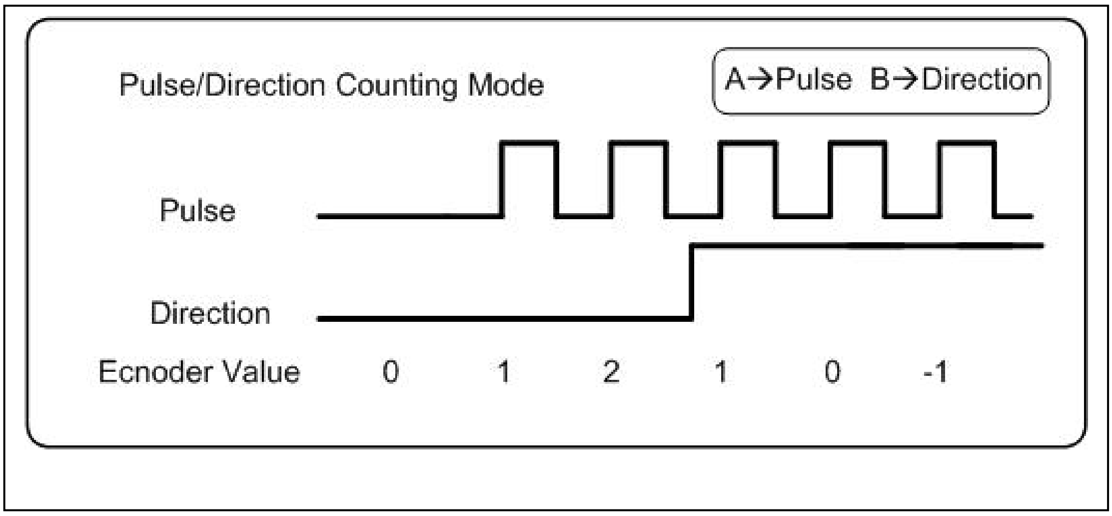
Refer to "I-7000 Bus Converter User Manual" chapter 1 for more information as following:

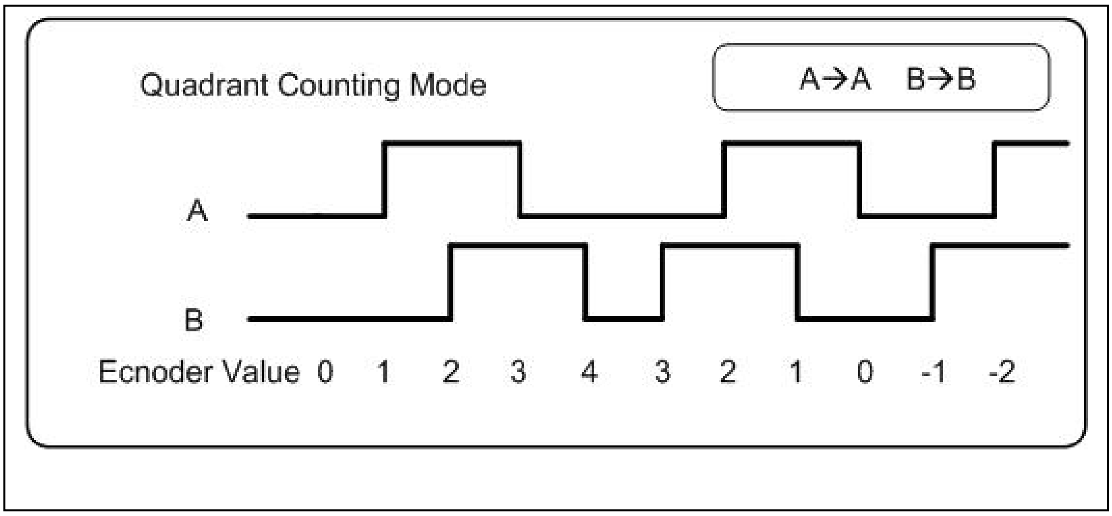
1.1 I-7000 Overview
1.2 I-7000 RELATED DOCUMENTATION
1.3 I-7000 COMMON FEATURES
1.4 I-7000 SYSTEM NETWORK CONFIGURATION
1.5 I-7000 Dimension

1.1 Encoder Counting Mode

There are 3 counting modes, CC/PD/AB, given as follows:



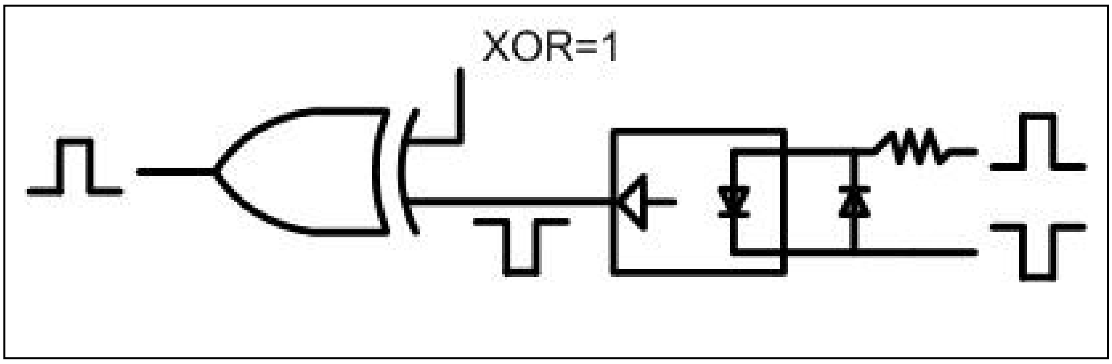




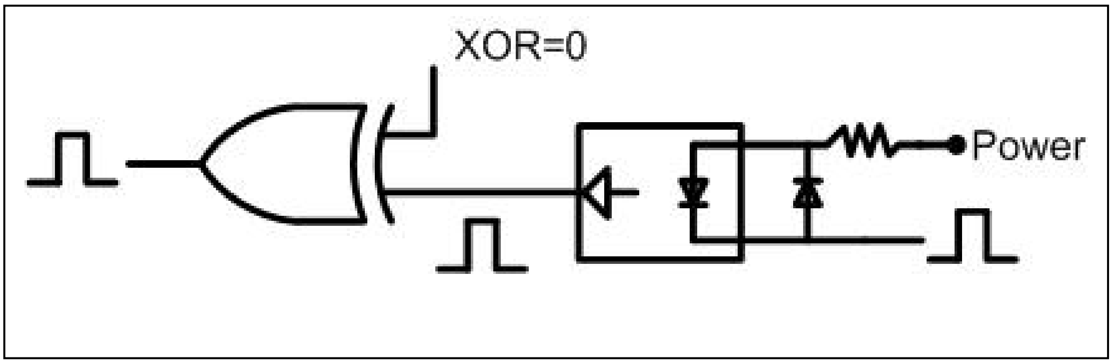
Note: -1=0xffffffff, -2=0xfffffffe

The internal counting logic is expected as **active high**. User can use XOR control bit to select the proper waveform as follows:

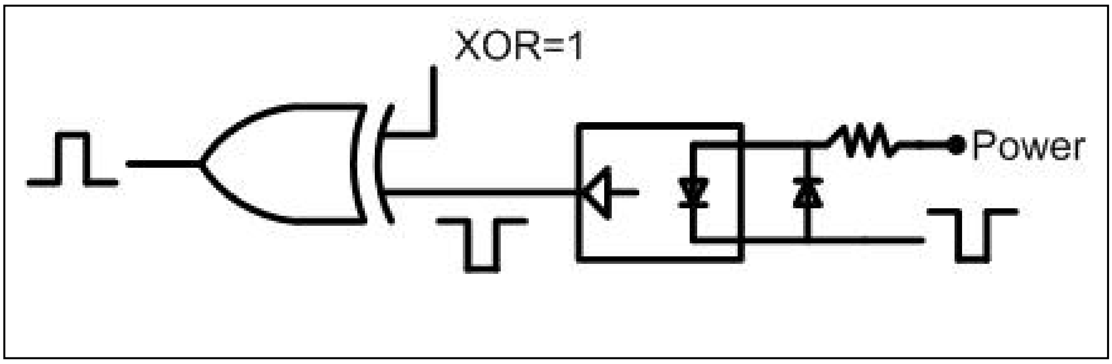
Case 1: differential input, set XOR=1



Case 2: active high single-ended input, set XOR=0



Case3: active low single-ended input, set XOR=1



If the value of XOR control bit is error, the encoder value will have different errors given as follows:

- The counting direction will be inverted
- The encoder value has error count = 1
- The Z is inverted

User can use $AASN command to check the status of A,B,Z. All A,B,Z are expected to be Low in the normal state & High in the active state. The check sequences are given as follows:

step	command	response
1	$01S0	!01M0
2	$01S1	!01M0
3	$01S2	!01M0

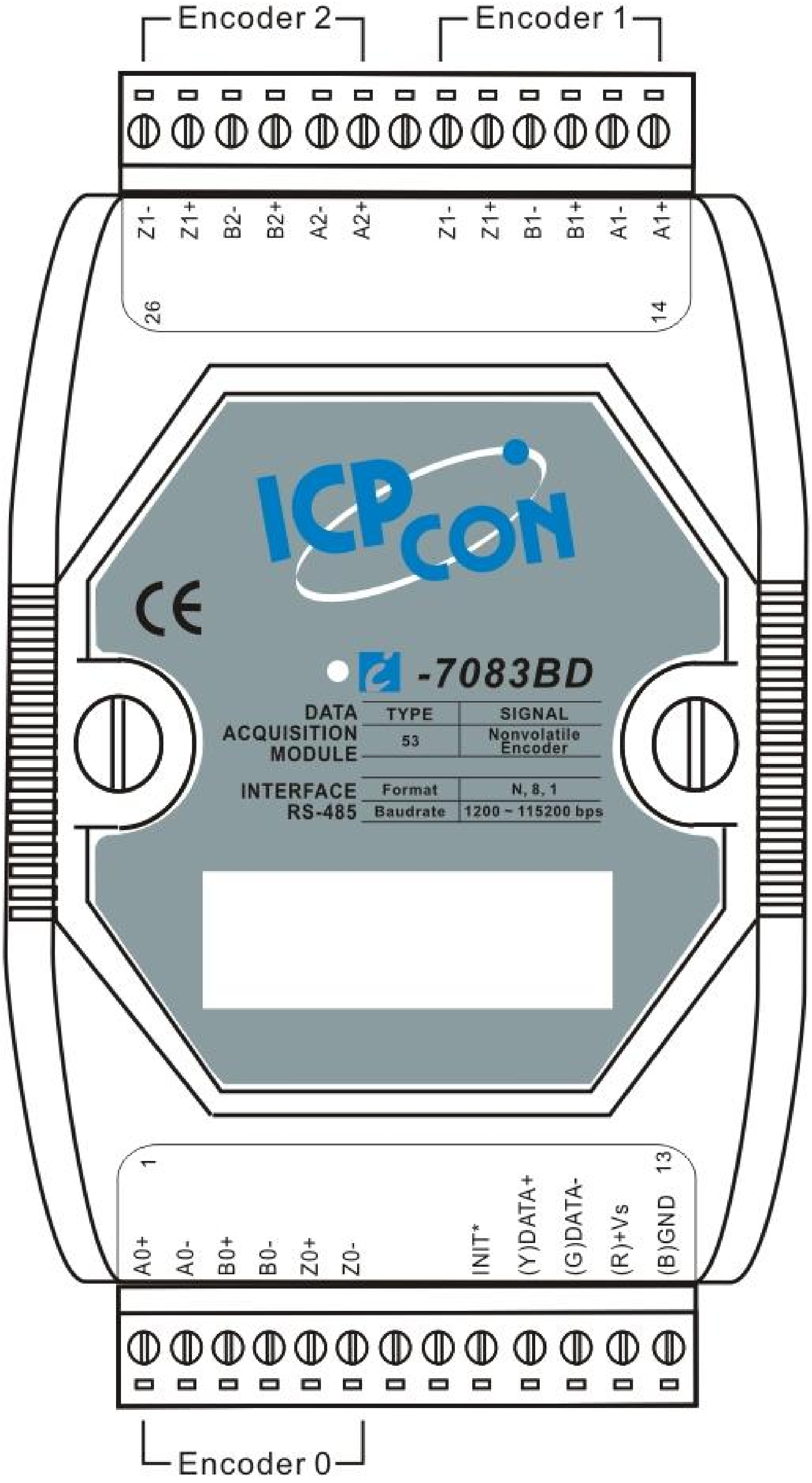
- Step 1: check A0,B0 & Z0 must be all Low, M=mode
- Step 2: check A1,B1 & Z1 must be all Low, M=mode
- Step 3: check A2,B2 & Z2 must be all Low, M=mode
- Refer to Sec. 2.18 for more information about $AASN command
- Refer to Sec. 1.6.6, Sec. 1.6.7 & Sec. 1.6.8 for more information about M=mode

Some error results are given as follows: (assume in the normal state)

step	command	response
1	$01S0	!01M7
2	$01S1	!01M7
3	$01S2	!01M7

- Step 1: XOR0 is setting error, A0,B0 & Z0 must be Low
- Step 2: XOR1 is setting error, A1,B1 & Z1 must be Low
- Step 3: XOR2 is setting error, A2,B2 & Z2 must be Low

1.2 Pin Assignment



1.3 Specifications

i-7083: 3-axis Encoder Module
i-7083D: i-7083 with LED Display
i-7083B: 3-axis Nonvolatile Encoder Module
i-7083BD: i-7083B with LED Display

Encoder Input

- Channels: Three independents 32 bit encoder counters, encoder 0,1,2
- Encoder Input:
  A1+,A1-,B1+,B1-,Z1+Z1- for encoder 0
  A2+,A2-,B2+,B2-,Z2+,Z2- for encoder 1
  A3+,A3-,B3+,B3-,Z3+,Z3- for encoder 2
- Encoder counting modes: Cw/Ccw, Pulse/Dir, A/B phase
- Input Level:
  Input 5V
    Logic High: 3.5~5V
    Logic Low: 0~2.0V
  Input 12V with external resistor, 1K ohm, 1/4W
    Logic High: 5~12V
    Logic Low: 0~2.0V
  Input 24V with external resistor, 2K ohm, 1/2W
    Logic High: 7~24V
    Logic Low: 0~2.0V
- Maximum counting rate: 1MHz
- A/B/Z signal isolation voltage: 2500V optical isolation
- Built-in XOR logic for active high or active low encoder input

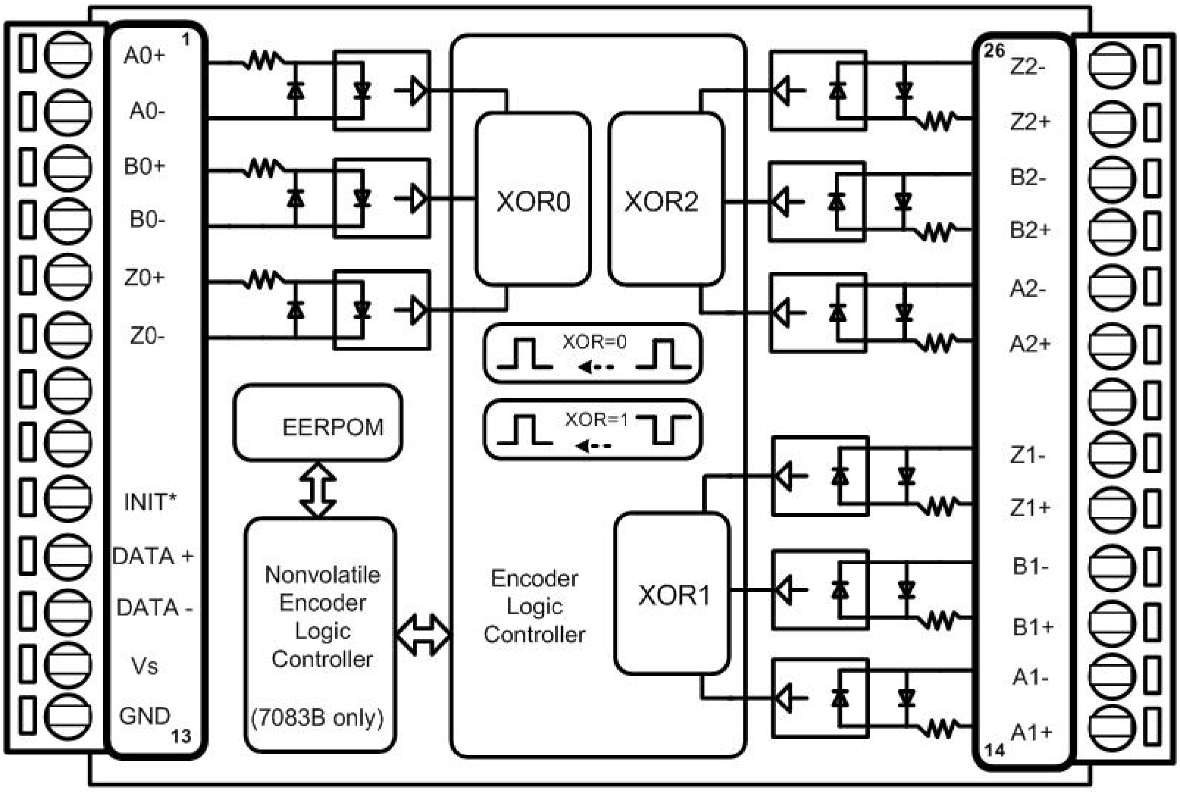
Display

- LED Indicator: 5-digit readout, channel 0 or channel 1

Power

- Power requirements: +10V to 30V(non-regulated)
- Power consumption : 1W for 7083, 7083B
  1.5W for 7083D, 7083BD

1.4 Block Diagram



	CW/CCW	Pulse/Direction	Quadrant
A	CW	Pulse	A
B	CCW	Direction	B

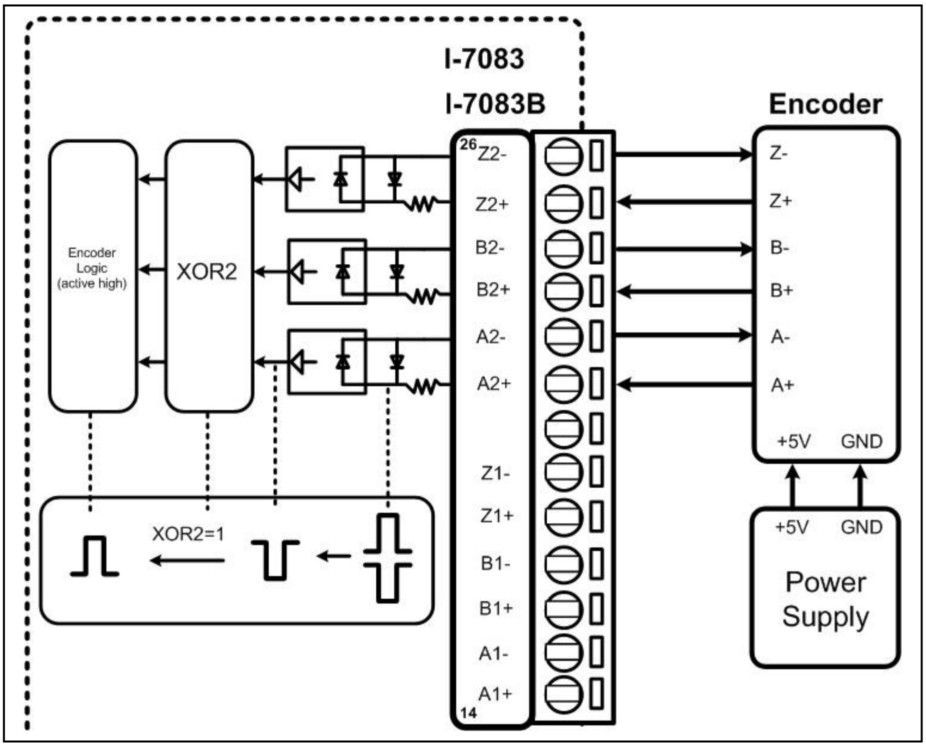
- Pin 1 ~ 6:A0+/A0-/B0+/B0-/ Z0+/Z0-, are designed for encoder0
- Pin 14 ~ 19:A1+/A1-/B1+/B1-/Z1+/Z1-, are designed for encoder1
- Pin 21 ~ 26:A2+/A2-/B2+/B2-/Z2+/Z2-, are designed for encoder2
- Pin 9 ~ 13: init*/DATA+/DATA-/Vs/GND, are same as 7000 series.

The input signal maybe active low or active high. The XOR0/XOR1/XOR2 are designed to invert the active low signal for internal logic requirement. If the value of XOR0/1/2 is the encoder value will have different errors. Refer to Sec. 1.1 for more information.

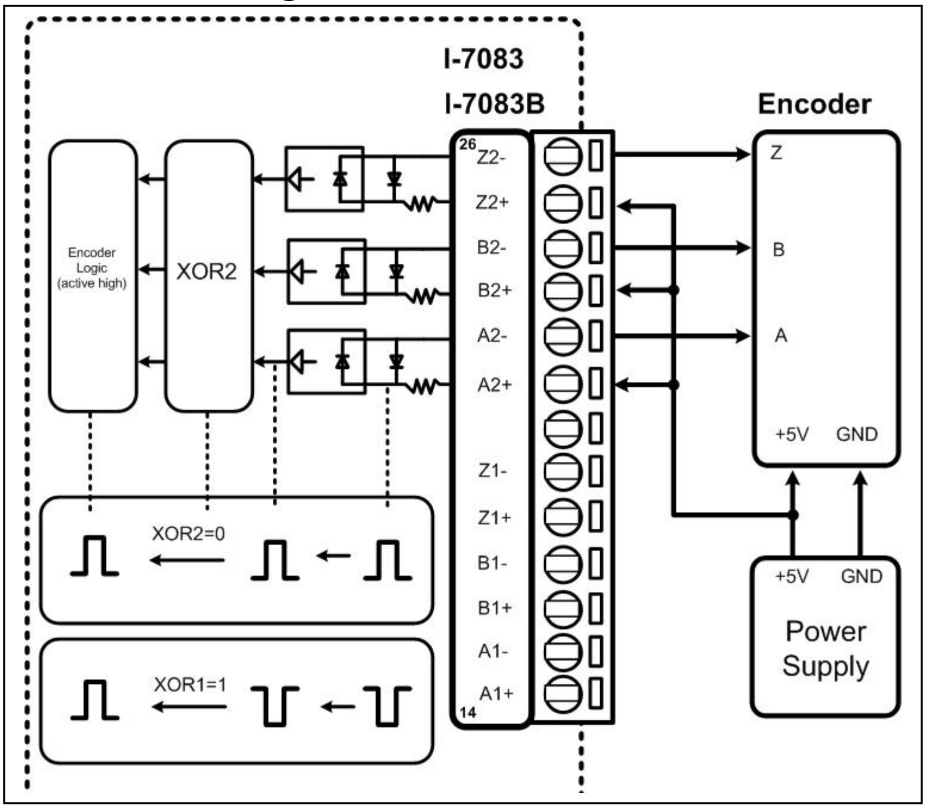
The 7083B equips a nonvolatile logic. When the power is turn OFF, all encoder values will be saved to all **preset values** in EEPROM. The 7083B will **re-store** all encoder values from all preset values when the power is turn ON. The 7083 does not equip this nonvolatile logic.

1.5 Application Wiring

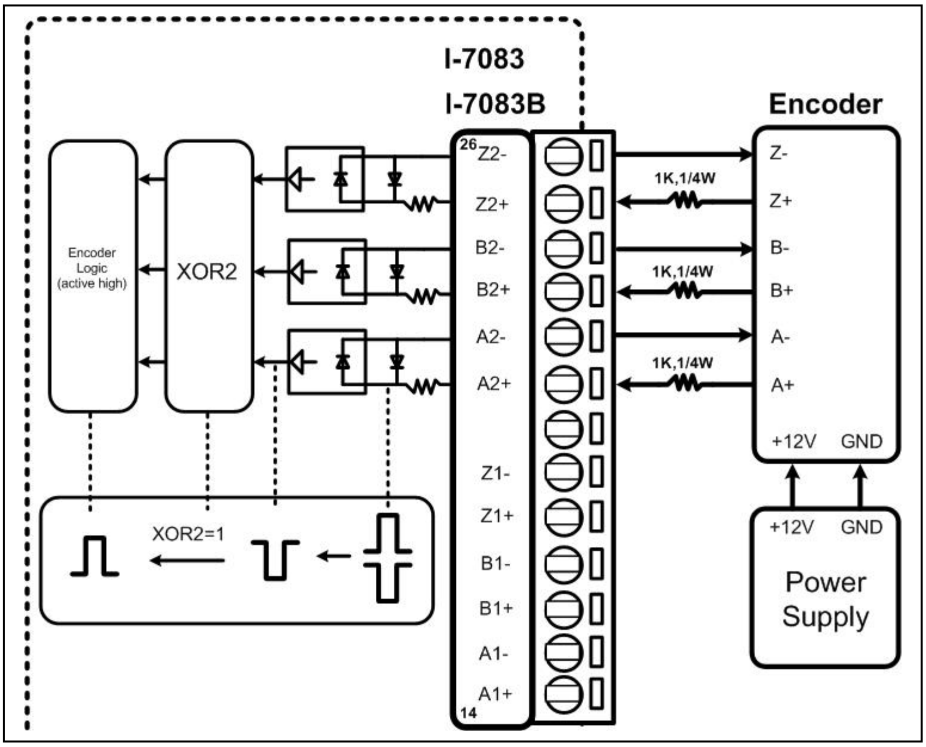
1.5.1 5V Differential Encoder



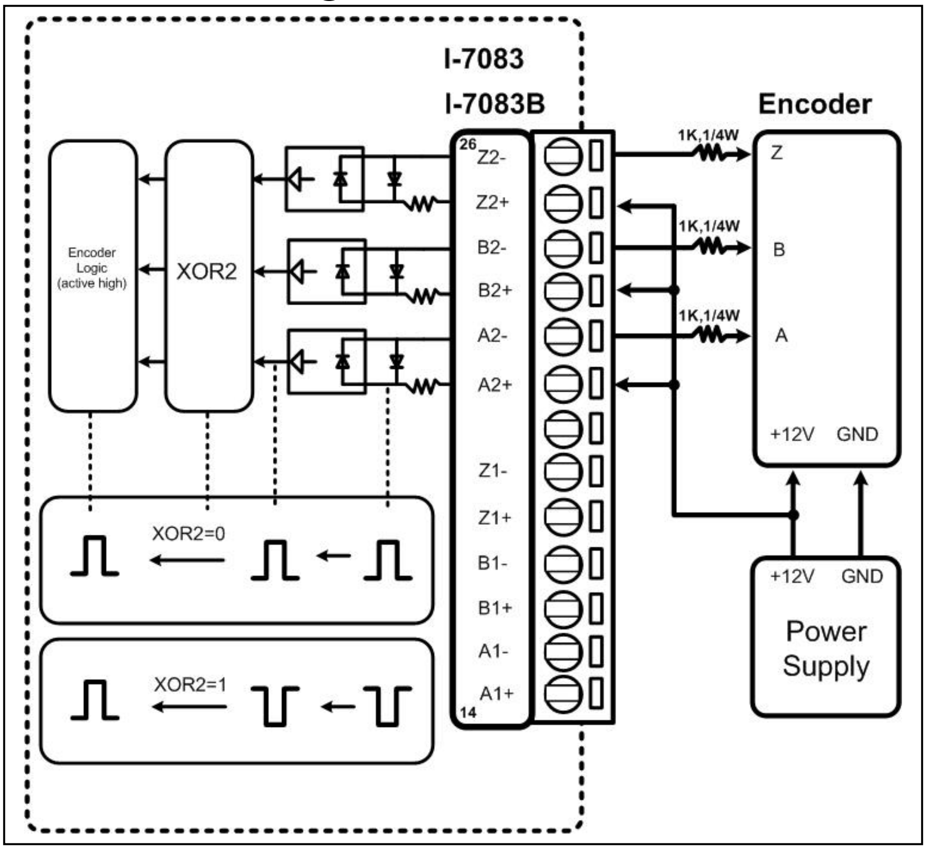
1.5.2 5V Single-ended Encoder



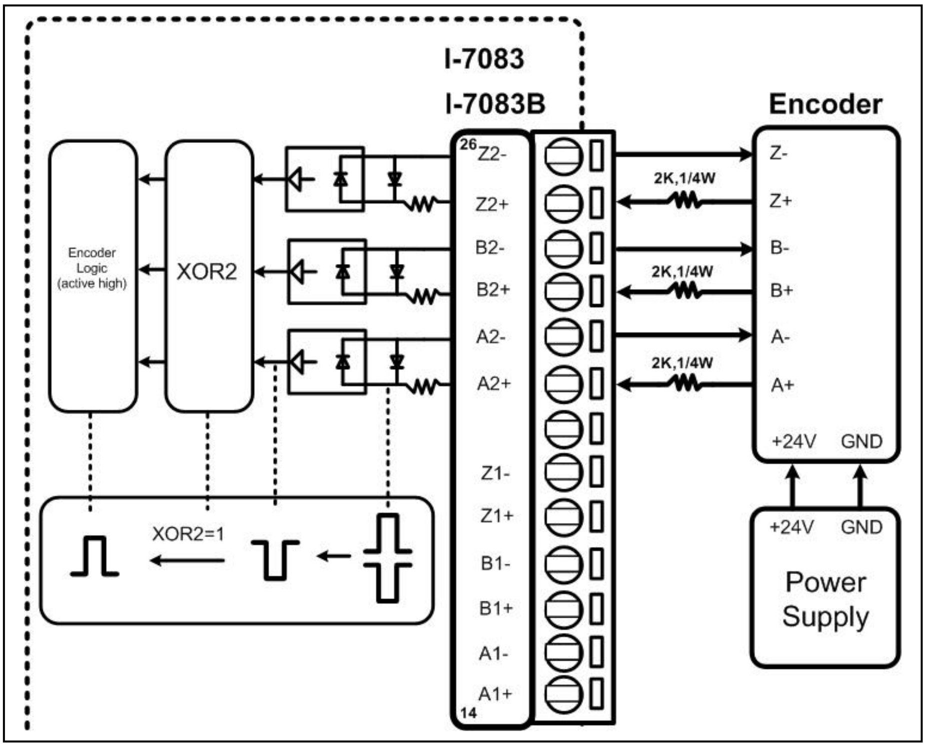
1.5.3 12V Differential Encoder



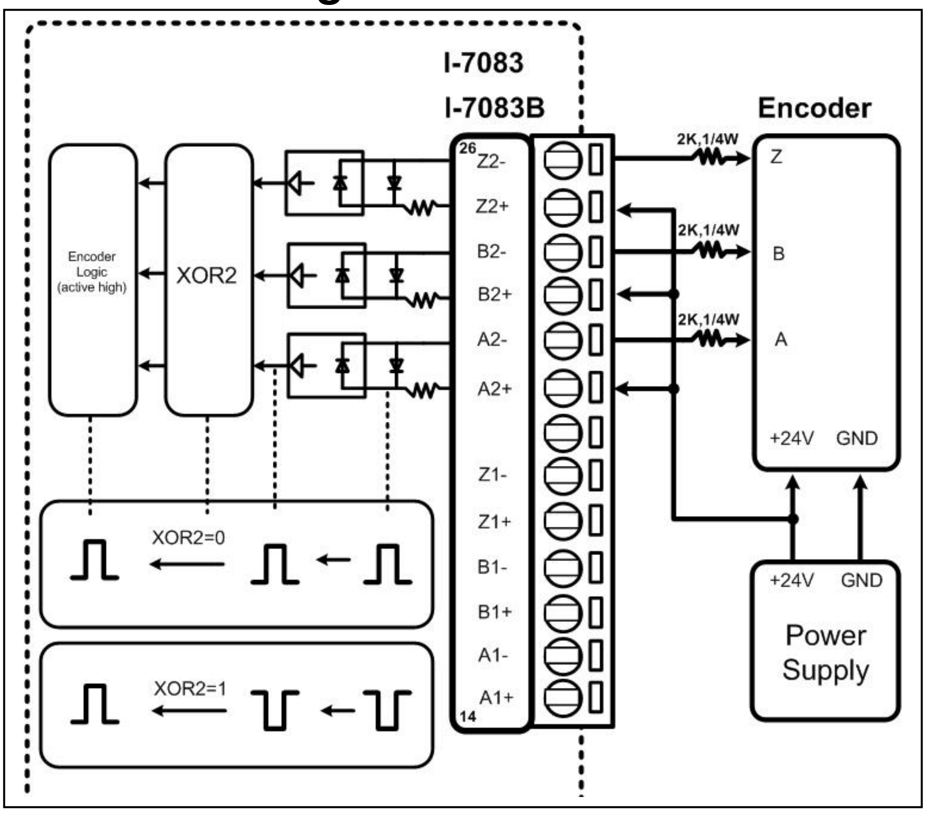
1.5.4 12V Single-ended Encoder



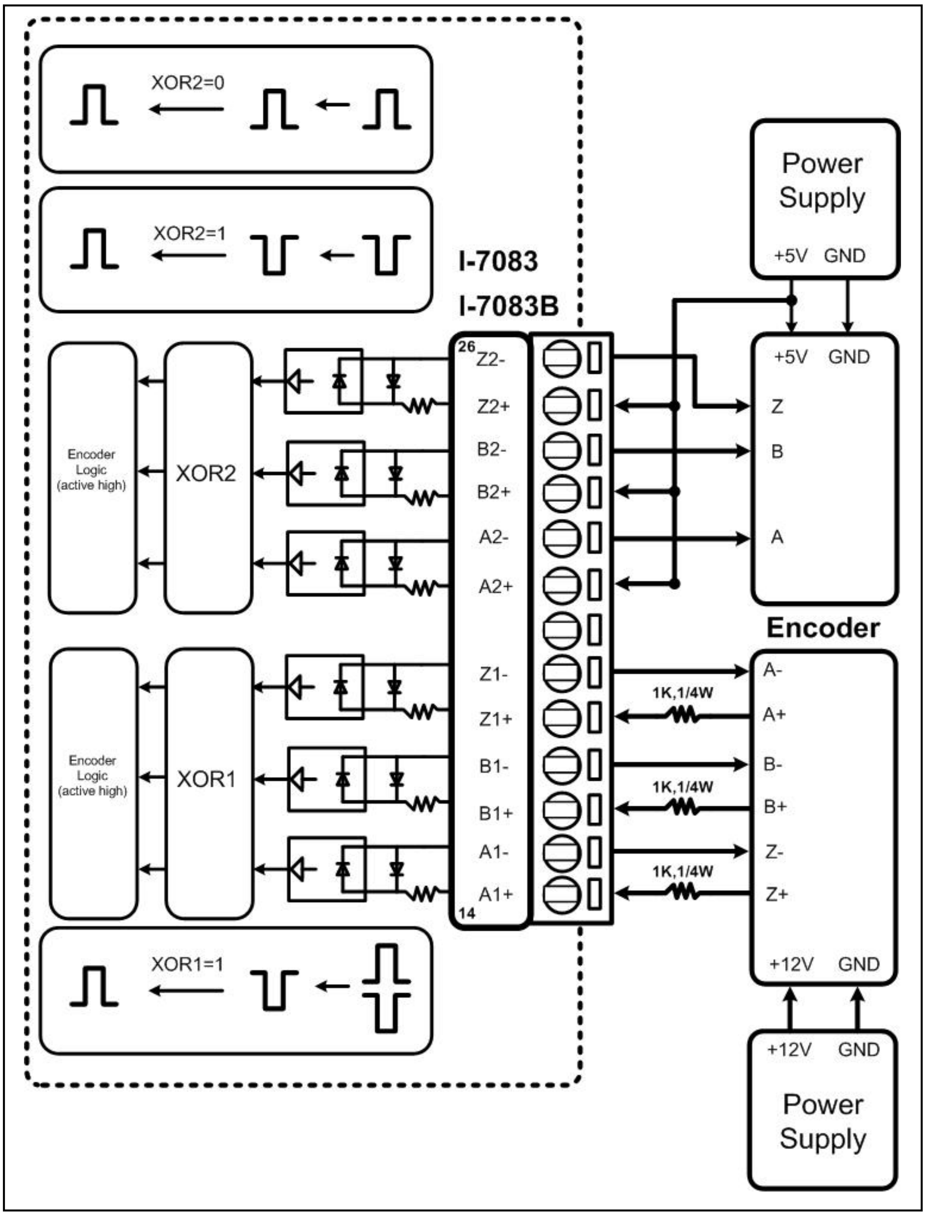
1.5.5 24V Differential Encoder



1.5.6 24V Single-ended Encoder



1.5.7 Mix-Mode Encoder



1.6 Quick Start

Assume the module address is 01 in the Sec. 1.6.

1.6.1 Read Encoder & Z2,Z1,Z0

step	command	response
1	$01M	!017083B
2	#010	>00000000
3	#011	>00000004
4	#012	>0000000A
5	$01S3	!0103

- step 1: read the module name, it is a 7083
- step 2: read encoder0
- step 3: read encoder1
- step 4: read encoder2
- step 5: read Z2,Z1&Z0, Z2=Low, Z1=Z0=High

1.6.2 Read Synchronous Encoder

step	Command	response
1	$01M	!017083
2	$02M	!027083B
3	#**	No Response
4	$01Z0	>00000000
5	$01Z1	>00000001
6	$01Z2	>00000002
7	$02Z0	>00000003
8	$02Z1	>00000004
9	$02Z2	>00000005

- step 3: synchronous latch all encoders
- step 4 ~ 6: read synchronous encoders of module 01
- step 7 ~ 9: read synchronous encoders of module 02

Note: all these 6 sync encoders are latched at the same time

1.6.3 Read Encoder & Synchronous Encoder

step	command	response
1	$01M	!017083
2	#010	>00000001
3	#011	>00000002
4	#012	>00000003
5	#**	No Response
6	$01Z0	>00000004
7	$01Z1	>00000004
8	$01Z2	>00000004
9	#010	>00000005
10	#011	>00000006
11	#012	>00000007
12	$01Z0	>00000004
13	$01Z1	>00000004
14	$01Z2	>00000004
15	#010	>00000009
16	#011	>0000000A
17	#012	>0000000B
18	#**	No Response
19	$01Z0	>0000000C
20	$01Z1	>0000000C
21	$01Z2	>0000000C

- step 2 ~ 4: all encoders are continuous counting
- step 5: synchronous latch all encoders
- step 6 ~ 8: read all synchronous encoders
- step 9 ~ 11: all encoders are continuous counting
- step 12 ~ 14: all synchronous encoders are not changed
- step 15 ~ 17: all encoders are continuous counting
- step 18: synchronous latch all encoders (new)
- step 19 ~ 21: read all synchronous encoders (new)

Note: 1. encoders will always counting
2. synchronous encoders will latch until # command**

1.6.4 Set the Preset Value of Encoder

step	command	response
1	$01M	!017083
2	@01G0	!0100000000
3	@01G1	!0100000000
4	@01G2	!0100000000
5	@01P010000000	!01
6	@01P120000000	!01
7	@01P230000000	!01

- step 2 ~ 4: read the preset value of encoder(default of 7083)
- step 5 ~ 7: set the preset value of encoder

Note:

1. For 7083, the preset value can be changed by @AAP(data) command. And the preset value will not change when the power is turn OFF.
2. For 7083B, the preset value can be changed by @AAP(data) command. But the encoder value will save to preset value when the power is turn OFF.

1.6.5 Clear Encoder to 0

step	command	response
1	$01M	!017083
2	@01P000000000	!01
3	@01P100000000	!01
4	@01P200000000	!01
5	$0160	!01
6	$0161	!01
7	$0160	!01

- step 2 ~ 4: set preset value = 0
- step 5 ~ 7: set encoder to preset value

1.6.6 Set Operation Mode to CwCcw Mode

step	command	response
1	$01M	!017083B
2	$01D01	!01
3	$01D15	!01
4	$01D2D	!01
5	$01S0	!0110
6	$01S1	!0150
7	$01S2	!01D0
8	$01S3	!0100

- step 2: set encoder 0 → CwCcw mode
  XOR0=0, input signal is single-ended & active high (Sec. 1.5.2)
  L0=0, the preset value no update when power if turn OFF

- step 3: set encoder 1 → CwCcw mode
  XOR1=1, input signal is differential (Sec. 1.5.1)
  L1=0, the preset value no update when power if turn OFF

- step 4: set encoder 2 → CwCcw mode
  XOR2=1, input signal is differential (Sec. 1.5.1)
  L2=1, encoder value will save to the preset value when power is turn OFF. It is the default setting of 7083B(D)

- step 5: mode read back=CwCcw, XOR0=0, L0=0,
  Z0=A0=B0=Low → Z,A,B must be normal Low & active High.

- step 6: mode read back=CwCcw, XOR1=1, L1=0,
  Z1=A1=B1=Low → Z,A,B must be normal Low & active High.

- step 7: mode read back=CwCcw, XOR2=0, L2=1,
  Z2=A2=B2=Low → Z,A,B must be normal Low & active High.

- step 8: read all Z, Z2=Z1=Z0=Low

1.6.7 Set Operation Mode to PulseDir Mode

step	command	response
1	$01M	!017083B
2	$01D02	!01
3	$01D16	!01
4	$01D2E	!01
5	$01S0	!0120
6	$01S1	!0160
7	$01S2	!01E0
8	$01S3	!0100

- step 2: set encoder 0 → PulseDir mode
  XOR0=0, input signal is single-ended & active high (Sec. 1.5.2)
  L0=0, the preset value no update when power if turn OFF

- step 3: set encoder 1 → PulseDir mode
  XOR1=1, input signal is differential (Sec. 1.5.1)
  L1=0, the preset value no update when power is turn OFF

- step 4: set encoder 2 → PulseDir mode
  XOR2=1, input signal is differential (Sec. 1.5.1)
  L2=1, encoder value will save to the preset value when power is turn OFF. It is the default setting of 7083B(D)

- step 5: mode read back=PulseDir, XOR0=0, L0=0,
  Z0=A0=B0=Low → Z,A,B must be normal Low & active High.

- step 6: mode read back=PulseDir, XOR1=1, L1=0,
  Z1=A1=B1=Low → Z,A,B must be normal Low & active High.

- step 7: mode read back=PulseDir, XOR2=0, L2=1,
  Z2=A2=B2=Low → Z,A,B must be normal Low & active High

- step 8: read all Z, Z2=Z1=Z0=Low

1.6.8 Set Operation Mode to A/B Phase Mode

step	command	response
1	$01M	!017083B
2	$01D03	!01
3	$01D17	!01
4	$01D2F	!01
5	$01S0	!0130
6	$01S1	!0170
7	$01S2	!01F0
8	$01S3	!0100

- step 2: set encoder 0 → A/B Phase mode
  XOR0=0, input signal is single-ended & active high (Sec. 1.5.2)
  L0=0, the preset value no update when power if turn OFF

- step 3: set encoder 1 → A/B Phase mode
  XOR1=1, input signal is differential (Sec. 1.5.1)
  L1=0, the preset value no update when power if turn OFF

- step 4: set encoder 2 → A/B Phase mode
  XOR2=1, input signal is differential (Sec. 1.5.1)
  L2=1, encoder value will save to the preset value when power is turn OFF. It is the default setting of 7083B(D)

- step 5: mode read back=A/B Phase, XOR0=0, L0=0,
  Z0=A0=B0=Low → Z,A,B must be normal Low & active High.

- step 6: mode read back=A/B Phase, XOR1=1, L1=0,
  Z1=A1=B1=Low → Z,A,B must be normal Low & active High.

- step 7: mode read back=A/B Phase, XOR2=0, L2=1,
  Z2=A2=B2=Low → Z,A,B must be normal Low & active High.

- step 8: read all Z, Z2=Z1=Z0=Low

1.7 Default Setting

The default setting is given as following:

- address=01
- baud rate=9600
- checksum disable
- data=1 start+8 data+1 stop(no parity)
- type=53
- Mode= 5 for 7083, L=0, XOR=1, CwCcw mode
  D for 7083B, L=1, XOR=1, CwCcw mode
  Refer to Sec. 1.6.6 for more information

1.8 Application Notes

1.8.1 Encoder & Synchronous Encoder

Encoder will always counting. Synchronous encoder will latch until next #** command is received.

User must read encoder & synchronous encoder one by one. So there is a time delay between each read operation. When host computer send #** command to RS-485 network, all 7083/7083B in this RS485 network will latch their synchronous encoders at the same time. Then host computer can read these synchronous encoders one by one. Refer to Sec. 1.6.2 & Sec. 1.6.3 for more information

1.8.2 Preset Value of Encoder

The @AAPN(data) can be used to set the preset value of encoder. The preset value is saved in the EEPROM. When the power is turn ON, the preset value will be loaded from EEPROM and set to the start value of encoder.

For 7083B, the current encoder value will save to the preset value in EEPROM when power is turn OFF. When the power is next turn ON, the preset value will be re-load from EEPROM. That it to say, the encoder value is **nonvolatile** even if the power is OFF. 7083B can set L-bit to 0 to disable the **nonvolatile logic**, Refer to Sec 1.6.6 for more information.

For 7083, there is no **nonvolatile logic,** so the start value of encoder is always as same as the preset value in EEPROM. The L-bit of 7083 is don't care.

1.8.3 Encoder Counting Sequence

The encoder is a 32-bit Up/Down counter without overflow. The 0x00000000 will change to 0xffffffff if one down counting is received. The 0xffffffff will change to 0x00000000 if one up counting is received. There is no overflow condition.

1.8.4 XOR Control Bit Setting

The internal logic is designed for active high. So the XOR control bit should be set to 1 in most of application. If the input signal is single-ended & active high (Sec. 1.5.2), the XOR bit has to be set to 0 for proper operation.

If the XOR bit is setting error, the encoder value will have different errors. Refer to Sec. 1.1 for more information.

1.9 Tables

Configuration Code Table : CC

CC	Baud Rate
03	1200 BPS
04	2400 BPS
05	4800 BPS
06	9600 BPS
07	19200 BPS
08	38400 BPS
09	57600 BPS
0A	115200 BPS

Configuration Code : FF, 2-char (for all)

7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	checksum 0=disable 1=enable	0					

Configuration Code Table: TT

TT	Input Range
53	Encoder

2. Command Set

General Command Set

Command	Response	Description	Reference
%AANNTTCCFF	!AA	Set module configuration	Sec. 2.1
#AAN	>(data)	Read encoder	Sec. 2.2
#**	No Response	Synchronous read encoder	Sec. 2.3
~**	No Response	Host OK	Sec. 2.4
~AA0	!AASS	Read module status	Sec. 2.5
~AA1	!AA	Reset module status	Sec. 2.6
~AA2	!AATT	Read host watchdog timer	Sec. 2.7
~AA3ETT	!AA	Enable host watchdog timer	Sec. 2.8
~AAM	!AA(data)	Read OEM module name	Sec. 2.9
~AAO(name)	!AA	Set module name	Sec. 2.10
$AA2	!AATTCCFF	Read configuration	Sec. 2.11
$AA5	!AAS	Read reset status	Sec. 2.12
$AA6N	!AA	Reset to the preset value	Sec. 2.13
$AADNM	!AA	Set mode of encoder	Sec. 2.14
$AAF	!AA(data)	Read firmware number	Sec. 2.15
$AAI	!AAS	Read the value of INIT* pin	Sec. 2.16
$AAM	!AA(data)	Read the module name	Sec. 2.17
$AASN	!AASS	Read status of encoder	Sec. 2.18
$AAZN	!AA(data)	Read the sync encoder	Sec. 2.19
@AAGN	!AA(data)	Read the preset value	Sec. 2.20
@AAPN(data)	!AA	Set the preset value	Sec. 2.21

2.1 %AANNTTCCFF

- **Description**: Set the configuration of module.

- **Syntax**: %AANNTTCCFF[chk](cr)
  % is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  NN=new AA
  TT=input type code, refer to Sec. 1.9
  CC=baud rate code, refer to Sec. 1.9
  FF=status code, refer to Sec. 1.9
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Response**: valid command → !AA[chk](cr)
  invalid command → ?AA[chk](cr)
  no response → syntax error or address error or communication error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Example**:
  command: %0102530600(cr)
  response : !02(cr)

  address 01 is configured to a new address 02, 9600BPS

  command: %0202530600(cr)
  response : !02(cr)

  Change to 9600BPS.

Refer to "I-7000 Bus Converter User Manual" chapter-5 for the following functions:

- **module status unknown**(Sec. 5.1), **change address**(Sec. 5.2)
- **change baud rate**(Sec. 5.3), **checksum enable/disable**(Sec. 5.4)

2.2 #AAN

- **Description**: Read encoder value.

- **Syntax**: #AAN[chk](cr)
  # is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  N=0 → encoder 0
  1 → encoder 1
  2 → encoder 2
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Response**:
  valid command → >[chk](data)(cr)
  invalid command → No Response
  no response → syntax error or communication error or address error
  > is a delimiter character indicating a valid command
  (data) = 8-character data(in HEX format)
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Example**:

command: $012(cr)
response : !01530600(cr)
command: #010(cr)
response : >0000001E(cr)

encoder-0=0x1E

command: $022(cr)
response : !02530600(cr)
command: #021(cr)
response : >8000001E(cr)

encoder-1=0x8000001E

2.3 #**

- **Description**: Synchronous read encoder.

- **Syntax**: #**[chk](cr)
  #　　　a delimiter character
  **　　　synchronous read command
  [chk]=2-charcter checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Response**: no response

- **Example**:

  command: $012(cr)
  response : !01530600(cr)

  command: #**(cr)
  response : No response

  Synchronous latch the encoder 0/1/2

  command: $01Z0(cr)
  response : >00000030(cr)

  Sync encoder 0=0x30

  command: $01Z1(cr)
  response : >00000031cr)

  Sync encoder 10x31

  command: $01Z2(cr)
  response : >00000032cr)

  Sync encoder 2=0x32

Refer to Sec. 1.8.1 for more information

2.4 ~**

- **Description**: Host send this command to tell all modules "Host is OK".

- **Syntax**: ~**[chk](cr)
  ~ is a delimiter character
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Response**: no response

- **Example**:
  command: ~**(cr)
  response : No Response

2.5 ~AA0

- **Description**: Read the module status. The module status will be latched until ~AA1 command is sent. **If the host watchdog is enable and the host is down, the module status will be set to 4. If the module status=4, all output command will be ignored.**

- **Syntax**: ~AA0[chk](cr)
  ~ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Response**: valid command → !AASS[chk](cr)
  invalid command → ?AA[chk](cr)
  no response → syntax error or communication error or address error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address
  SS=2-character HEX status value as following:
  Bit_0, Bit_1 = reserved
  Bit_2 = 0 → OK,
  1 → host watchdog failure
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Example**:

command: ~010(cr)
response : !0100(cr)

Status of module 01 is OK

command: ~020(cr)
response : !0204(cr)

Module status=04 → host watchdog failure → HOST is down now

2.6 ~AA1

- **Description**: Reset the module status. The module status will be latched until ~AA1 command is sent. If the module status is not 0, only ~AA1 command can clear the module status.

- **Syntax**: ~AA1[chk](cr)
  ~ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D
- **Response**: valid command → !AA[chk](cr)
  invalid command → ?AA[chk](cr)
  no response → syntax error or communication error or address error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Example**:

command: ~010(cr)
response : !0104(cr)

module status=0x04 → host is down

command: ~011(cr)
response : !01(cr)

clear module status

command: ~010(cr)
response : !0100(cr)

module status=0x00

2.7 ~AA2

- **Description**: Read the status and timer value of host watchdog. The host watchdog timer is designed for host watchdog. When the host watchdog is enable, the host must send ~** command to all modules before the timer is up. When the ~** command is received, the host watchdog timer is reset and restart. Use ~AA3ETT to enable/disable/setting the host watchdog timer.

- **Syntax**: ~AA2[chk](cr)
  ~ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D
- **Response**: valid command → !AASTT[chk](cr)
  invalid command → ?AA[chk](cr)
  no response → syntax error or communication error or address error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address
  S=0: host watchdog is disable
  S=1: host watchdog is enable
  TT=2-character HEX value, from 00 to FF, unit=0.1 second
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Example**:

  command: ~012(cr)
  response : !01000(cr)

  Host watchdog timer of module 01 is disable

  command: ~022(cr)
  response : !0210A(cr)

  host watchdog timer of module 02 is enable and = 0.1*10 = 1 second.

2.8 ~AA3ETT

- **Description**: Enable/disable the timer value of host watchdog. The host watchdog timer is designed for software host watchdog. When the software host watchdog is enable, the host must send ~** command to all modules before the timer is up. When the ~** command is received, the host watchdog timer is reset and restart. Use ~AA2 to read the host watchdog status & value.

- **Syntax**: ~AA3ETT[chk](cr)
  ~ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  E=0 is disable and 1 is enable
  TT=2-character HEX value, from 00 to FF, unit=0.1 second
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Response**: valid command → !AA[chk](cr)
  invalid command → ?AA[chk](cr)
  no response → syntax error or communication error or address error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Example**:
  command: ~013000(cr)
  response : !01(cr)

  disable host watchdog timer of module 01

  command: ~02310A(cr)
  response : !02(cr)

  host watchdog timer of module 02 is enable and = 0.1*10 = 1 second.

2.9 ~AAM

- **Description**: Read the OEM module name.

- **Syntax**: ~AAM[chk](cr)
  ~ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Response**: valid command → !AA(data)[chk](cr)
  invalid command → ?AA[chk](cr)
  no response → syntax error or communication error or address error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address
  data=4-character for module name
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Example**:

command: ~01M(cr)	response : !0100007083(cr)	OEM module name of 01 is 00007083
command: ~02M(cr)	response : !0200007083D(cr)	OEM module name of 02 is 00007083D

2.10 ~AAO(name)

- **Description**: Set module name.

- **Syntax**: ~AAO(name)[chk](cr)
  ~ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  (name)=4-character/5-character module name
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Response**: valid command → !AA[chk](cr)
  invalid command → ?AA[chk](cr)
  no response → syntax error or communication error or address error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Example**:
  command: $01M(cr)
  response : !017083(cr)
  command: ~01O8083(cr)
  response : !01(cr)

  Change module name from 7083 to 8083

  command: $01M(cr)
  response : !017083D(cr)
  command: ~01O8083D(cr)
  response : !01(cr)

  Change module name from 7083D to 8083D

Note: This command is designed for OEM/ODM user. The user can use it to change the module name for other purpose.

2.11 $AA2

- **Description**: Read the configuration of module.
- **Syntax**: $AA2[chk](cr)
  $ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D
- **Response**: valid command → !AATTCCFF[chk](cr),
  invalid command → ?AA[chk](cr)
  no response → syntax error or communication error or address error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address
  TT, CC, FF: refer to Sec. 1.9
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D
- **Example**:
  command: $012(cr)
  response : !01530600(cr)

  Address=01, encoder, 9600 BPS, checksum disable

  command: $022(cr)
  response : !02530700(cr)

  Address=02, encoder, 19200 BPS, checksum disable

NOTE: If the user use %AANNTTCCFF command to change module configuration, the new configuration code will be stored into EEPROM immediately. The configuration code includes module address, module type, baud rate code, checksum enable/disable code, calibration code, power-on value and safe value. **The EEPROM data of I-7000 can be read infinite times and can be written about 100,000 times max.** Therefore the user should not change configuration code often for testing.
The $AA2 command is used to read EEPROM data only, therefore the user can send this command to I-7000 module infinitely.

2.12 $AA5

- **Description**: Read reset status
- **Syntax**: $AA5[chk](cr)
  $ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Response**: valid command → !AAS[chk](cr)
  invalid command → ?AA[chk](cr)
  no response → syntax error or communication error or address error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address
  S=1, this module is been reset
  S=0, this module is not been reset
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Example**:

command: $015(cr)
response : !011(cr)

Reset status=1, first read

command: $015(cr)
response : !010(cr)

Reset status=0, second read

2.13 $AA6N

- **Description**: Reset encoder to the preset value. Refer to Sec. 1.6.4 & Sec. 1.6.5 for more information.

- **Syntax**: $AA6N[chk](cr)
  $ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  N=0 → encoder 0
  1 → encoder 1
  2 → encoder 2
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Response**: valid command → !AA[chk](cr)
  invalid command → ?AA[chk](cr)
  no response → syntax error or communication error or address error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Example**:

command: @01G0(cr)
response : !0100000000(cr)
command: $0160(cr)
response : !01(cr)

Preset value=0
Reset encoder 0 to preset value 0x00000000

command: @01G1(cr)
response : !010000ABCD(cr)
command: $0161(cr)
response : !01(cr)

Preset value=0xABCD
Reset encoder 1 to preset value 0x0000ABCD

2.14 $AADNM

- **Description**: Set the operation mode of encoder. Refer to Sec. 1.6.6 for more information.

- **Syntax**: $AADNM[chk](cr)
  $ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  N=0/1/2 → encoder 0/1/2
  M=LXCC → CC=00/01/10/11=stop/UD/DP/AB
  X=0/1=XOR control bit
  L=1,update the preset value before power is off
  L=0,no update the preset value
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Response**: valid command → !AA[chk](cr)
  invalid command → ?AA[chk](cr)
  no response → syntax error or communication error or address error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Example**:

command: $01D05(cr)
response : !01(cr)

Up/Down counting mode(UD)
XOR control bit = 1
Preset value no update

command: $01D1B(cr)
response : !01(cr)

AB phase counting mode(AB)
Xor control bit = 0
Update the preset value before the power is turn off

2.15 $AAF

- **Description**: Read the version number of firmware.
- **Syntax**: $AAF[chk](cr)
  $ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D
- **Response**: valid command → !AA(data)[chk](cr)
  invalid command → ?AA[chk](cr)
  no response → syntax error or communication error or address error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address
  data=5-character for version number
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D
- **Example**:

command: $01F(cr)
response : !01A2.0(cr)

Ver. A2.0

command: $02F(cr)
response : !02A3.0(cr)

Ver. A3.0

2.16 $AAI

- **Description**: Read the value of *INIT pin.
- **Syntax**: $AAI[chk](cr)
  $ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Response**: valid command → !AAS[chk](cr)
  invalid command → ?AA[chk](cr)
  no response → syntax error or communication error or address error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address
  S=0 → INIT* pin is connected to GND pin
  1 → INIT* pin is open
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Example**:

command: $01I(cr)
response : !010(cr)

INIT* pin is connected to GND pin.

command: $02I(cr)
response : !021(cr)

INIT* pin is open.

2.17 $AAM

- **Description**: Read the module name.
- **Syntax**: $AAM[chk](cr)
  $ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D
- **Response**: valid command → !AA(data)[chk](cr)
  invalid command → ?AA[chk](cr)
  no response → syntax error or communication error or address error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address
  data=4-character for module name
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D
- **Example**:

command: $01M(cr)
response : !017083(cr)

Module name of 01 is 7083

command: $02M(cr)
response : !027083D(cr)

Module name of 02 is 7083D

2.18 $AASN

- **Description**: Read status of encoder. Refer to Sec. 1.6.6 for more information.

- **Syntax**: $AASN[chk](cr)
  $ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Response**: valid command → !AASS[chk](cr)
  invalid command → ?AA[chk](cr)
  no response → syntax error or communication error or address error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address

(N=0/1/2)→SS=LXCC 0ZBA → CC=00/01/10/11=stop/UD/DP/AB
X=0/1=XOR control bit
L=0/1,preset value update
(N=3) →SS=0000ZZZ → bit2=Z2, bit1=Z1, bit0=Z0 → Sec. 1.6.1
[chk]=2-character checksum, if checksum disable → no [chk]
(cr)=0x0D

- **Example**:

command: $01S0(cr)
response : !0150(cr)

Up/Down counting mode(UD)
XOR control bit = 1
Preset value no update

command: $01S3(cr)
response : !0105(cr)

Z2=High
Z1=Low
Z0=High

2.19 $AAZN

- **Description**: Read the synchronous encoder value. Refer to Sec. 1.6.3 for more information

- **Syntax**: $AAZN[chk](cr)
  $ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  N=0 → channel-0 of encoder
  1 → channel-1 of encoder
  2 → channel-1 of encoder
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Response**:
  valid command → >[chk](data)(cr)
  invalid command → No Response
  no response → syntax error or communication error or address error
  > is a delimiter character indicating a valid command
  (data) = 8-character data(in HEX format)
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Example**:

  command: #**(cr)
  response : No Response

  command: $01Z0(cr)
  response : >0000001E(cr)

  Sync encoder 0 = 0x1E

  command: $01Z1(cr)
  response : >0000001F(cr)

  Sync encoder 1 = 0x1F

2.20 @AAGN

- **Description**: Read the preset value of counter. The $AA6N command can reset counter to the preset value. Refer to Sec. 1.8.2 for more information.

- **Syntax**: @AAGN[chk](cr)
  @ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  N=0 → read counter 0
  1 → read counter 1
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D
- **Response**: valid command → !AA(data)[chk](cr)
  invalid command → ?AA[chk](cr)
  no response → syntax error or communication error or address error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address
  (data)=8-character HEX value.
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Example**:
  command: @01G0(cr)
  response : !010000FFFF(cr)

  The preset value of encoder 0 is 0000FFFF.

  command: @02G1(cr)
  response : !0200000000(cr)

  The preset value of encoder 1 is 00000000.

2.21 @AAPN(data)

- **Description**: Set the preset value of counter. The $AA6N command can reset counter to preset value. Refer to Sec. 1.8.2 for more information.

- **Syntax**: @AAPN(data)[chk](cr)
  @ is a delimiter character
  AA=2-character HEX module address, from 00 to FF
  (data)=8-character HEX value.
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D
- **Response**: valid command → !AA(data)[chk](cr)
  invalid command → ?AA[chk](cr)
  no response → syntax error or communication error or address error
  ! is a delimiter character indicating a valid command
  ? is a delimiter character indicating a invalid command
  AA=2-character HEX module address
  [chk]=2-character checksum, if checksum disable → no [chk]
  (cr)=0x0D

- **Example**:
  command: @01P0FFFF0000(cr)
  response : !01(cr)

The preset value of encoder 0 is FFFF0000.

  command: @02P10000FFFF(cr)
  response : !02(cr)

The preset value of encoder 1 is 0000FFFF.

3. Operation Principle

3.1 INIT* Pin

All I-7000 modules contain an EEPROM to store configuration information. Therefore the user is difficult to find out the status of the I-7000 modules. The user can connect the INIT*_pin to GND_pin and power on the module. The I-7000 modules will **go to the factory default setting without changing the EEPROM data.** The factory default setting is given as following:

address	= 00
baud rate	= 9600
checksum	= DISABLE
data format	= 1 start + 8 data bits + 1 stop bit

If the user disconnect the INIT*_pin and GND_pin, the I_7000 module will be auto configured according to the EEPROM data. The user is easy to find the EEPROM configuration data in the default setting. The steps are shown as following:

Step 1: power off and connect INIT*_pin to GND_pin
Step 2: power on
Step 3: send command string **$002[0x0D]** to the module, the module will return back the EEPROM data.
Step 4: record the EEPROM data of this I-7000 module
Step 5: power off and disconnect INIT*_pin and GND_pin
Step 6: power on

Refer to "I-7000 Bus Converter User Manual" Sec. 5.1 for more information.

3.2 LED Display Format

The 7-Seg LED will show encoder 0/1/2 value one bye one as follows:

Step 1: **0.** + byte 8 + byte 7 + byte 6 + byte 5

Step 2: **0** + byte4 + byte 3 + byte 2 + byte 1

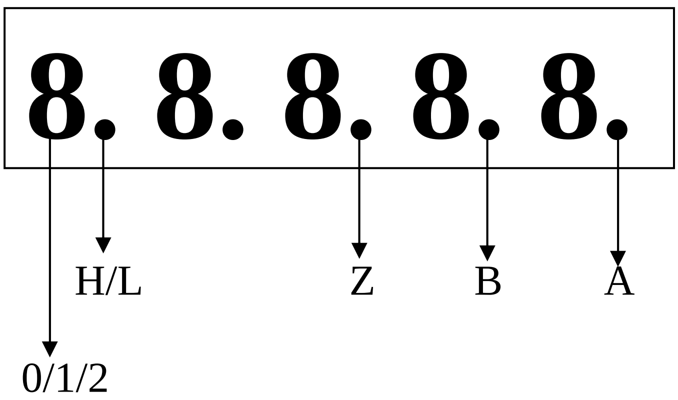
Step 3: **1.** + byte 8 + byte 7 + byte 6 + byte 5

Step 4: **1** + byte4 + byte 3 + byte 2 + byte 1

Step 5: **2.** + byte 8 + byte 7 + byte 6 + byte 5

Step 6: **2** + byte4 + byte 3 + byte 2 + byte 1

Note:
1. step 1 & step 2 will show encoder0
2. step 3 & step 4 will show encoder1
3. step 5 & step 6 will show encoder2
4. the decimal point of byte 5 & byte 1 is Hi/Lo status of A
5. the decimal point of byte 6 & byte 2 is Hi/Lo status of B
6. the decimal point of byte 7 & byte 3 is Hi/Lo status of Z



3.3 7080(D) & 7083B(D)

7080(D) & 7083B(D)

	7080(D)	7083B(D)
Standard version	7080, 7080D	7083, 7083D
Nonvolatile version	7080B, 7080BD	7083B, 7083BD
Counter length	32 bits	32-bits
Set preset value	@AAPN(data)	Same
Read preset value	@AAGN	Same
Read counter	#AAN	Same
Number of channels	2 channels	3 channels
Default setting	Up counting	CC(Up/Down counting)
Counting mode	Up counting	CC/PD/AB
Max. frequency	100K	1M
Synchronous Sampling	N/A	#**
Read sync encoder	N/A	$AAZN
Counter/Frequency	Programmable	Encoder only

7083(D) & 7083B(D)

	7083(D)	7083B(D)
Standard version	7083	7083B
With LED version	7083D	7083BD
Module type	53	53
Set L=0 (Sec. 2.14)	No effect	Preset value will not save. (Same as 7083(D))

7080(D) & 7080B(D)

	7080(D)	7080B(D)
Standard version	7080	7080B
With LED version	7080D	7080BD
Module type	50/51	50/51/52
Module type=50/51	Counter/Frequency	Same as 7080(D)
Module type=52	N/A	**Nonvolatile version of 50**